Mach Learn (2008) 71: 243–264
DOI 10.1007/s10994-008-5055-9

Efficient approximate leave-one-out cross-validation for kernel logistic regression

Gavin C. Cawley · Nicola L.C. Talbot

Received: 7 March 2007 / Revised: 24 March 2008 / Accepted: 29 March 2008 / Published online: 17 April 2008
Springer Science+Business Media, LLC 2008

Abstract Kernel logistic regression (KLR) is the kernel learning method best suited to binary pattern recognition problems where estimates of *a-posteriori* probability of class membership are required. Such problems occur frequently in practical applications, for instance because the operational prior class probabilities or equivalently the relative misclassification costs are variable or unknown at the time of training the model. The model parameters are given by the solution of a convex optimization problem, which may be found via an efficient iteratively re-weighted least squares (IRWLS) procedure. The generalization properties of a kernel logistic regression machine are however governed by a small number of hyper-parameters, the values of which must be determined during the process of model selection. In this paper, we propose a novel model selection strategy for KLR, based on a computationally efficient closed-form approximation of the leave-one-out cross-validation procedure. Results obtained on a variety of synthetic and real-world benchmark datasets are given, demonstrating that the proposed model selection procedure is competitive with a more conventional k-fold cross-validation based approach and also with Gaussian process (GP) classifiers implemented using the Laplace approximation and via the Expectation Propagation (EP) algorithm.

Keywords Model selection · Kernel logistic regression

1 Introduction

Kernel learning methods (see e.g. Müller et al. 2001; Schölkopf and Smola 2002; Shawe-Taylor and Cristianini 2004), such as the support vector machine (Boser et al. 1992; Cortes and Vapnik 1995; Vapnik 1998), kernel Fisher discriminant analysis (Mika et al. 1999, 2003), kernel ridge regression (Saunders et al. 1998) and kernel principal component analysis (Schölkopf et al. 1997; Mika et al. 2003), have attracted considerable interest

Editor: Olivier Chapelle.

G.C. Cawley (✉) · N.L.C. Talbot
School of Computing Sciences, University of East Anglia, Norwich, NR4 7TJ, UK
e-mail: gcc@cmp.uea.ac.uk

in the machine learning community in recent years, due to a combination of mathematical tractability and state-of-the-art performance demonstrated over a wide range of benchmark datasets (e.g. Cawley and Talbot 2003) and real-world applications (e.g. Brown et al. 2000). Kernel learning methods generally aim to construct a linear model in a feature space induced by a positive definite Mercer kernel (Mercer 1909). Depending on the choice of kernel, the feature space may be of high or even infinite dimension, while dealing with only finite dimensional quantities, such as the kernel matrix giving the value of the kernel function for every pair of data points comprising the training sample. The richness of the feature space allows the construction of very complex, powerful models, however Tikhonov regularization (Tikhonov and Arsenin 1977) has proved an effective means of capacity control, preventing over-fitting and optimizing generalization. The linear nature of the underlying model means that the optimal model parameters are often given by the solution of a convex optimization problem (Boyd and Vandenberghe 2004), with a single global optimum, for which efficient algorithms exist. The generalization properties of kernel methods however tend to be heavily dependent on the values of a small number of *hyper-parameters*, including regularization parameters (Tikhonov and Arsenin 1977) and parameters defining the kernel function (Chapelle et al. 2002). The search for the optimal values of these hyper-parameters is a process known as *model selection*. Unfortunately the model selection criteria for kernel learning methods are not generally unimodal, and so this paper is concerned with efficient search methods for finding a *locally* optimal set of hyper-parameter values.

The most common approach to model selection aims to minimize some form of cross-validation (Stone 1974) estimate of an appropriate model selection criterion, for example the misclassification rate or perhaps the cross-entropy in the case of a statistical pattern recognition problem. Under a k-fold cross-validation scheme, the available data are divided into k disjoint subsets. A model is then trained on $k-1$ of these subsets and the model selection criterion evaluated on the unused subset. This procedure is then repeated for all k combinations of $k-1$ subsets. The k-fold cross-validation estimate for the model selection criterion is then simply the mean of the model selection criterion computed over the unused subset in each fold. Cross-validation makes good use of the available data as all data are used as both training and test data. The most extreme form of k-fold cross-validation, in which each subset consists of a single training pattern is known as *leave-one-out* cross-validation (Lachenbruch and Mickey 1968). An attractive property of leave-one-out cross-validation for model selection purposes is that it provides an almost unbiased estimate of generalization performance (Luntz and Brailovsky 1969). The regularization and kernel parameters can then be tuned via minimization of the leave-one-out error using standard optimization techniques, such as the Nelder-Mead simplex algorithm.

Unlike kernel methods based on a least-squares training criterion, exact leave-one-out cross-validation of kernel logistic regression cannot be performed efficiently in closed-form. However in this paper, we propose a useful approximation, based on exact leave-one-out cross-validation of the quadratic approximation to the regularized training criterion obtained during the final iteration of the IRWLS training algorithm. This extends an existing efficient leave-one-out method for kernel Fisher discriminant analysis (Cawley and Talbot 2003) to be adapted for computationally efficient model selection for a family of kernel learning methods, including kernel logistic regression. The approximation is also shown to be equivalent to taking a single Newton step to minimize the reduced training criterion in each iteration of the leave-one-out procedure, starting from the optimal parameters for a model fitted to the entire training sample (see Appendix A).

The remainder of this paper is structured as follows: Sect. 2 introduces the kernel logistic regression model and introduces the notation used throughout. Section 3 proposes a simple model selection procedure for KLR based on an efficient, closed-form approximation of the leave-one-out and k-fold cross-validation estimates of the test cross-entropy. Section 4 compares model selection procedures based on approximate leave-one-out and conventional k-fold cross-validation. Results are also obtained for expectation-propagation based Gaussian process classifiers, providing a state-of-the-art baseline for comparison purposes. Results presented in Sect. 4 demonstrate that the approximate leave-one-out cross-validation procedure is competitive with the alternative procedures, at a vastly reduced computationally expense. Finally, the work is summarized and conclusions drawn in Sect. 5.

2 Kernel logistic regression

In this section, we provide a brief overview of the kernel logistic regression (KLR) model, and introduce the notation used throughout. In an off-line statistical pattern recognition problem, we are given labeled training data,

$$\mathcal{D} = \{(\mathbf{x}_i, t_i)\}_{i=1}^{\ell}, \qquad \mathbf{x}_i \in \mathcal{X} \subset \mathbb{R}^d, \; t_i \in \{0, \; 1\},$$

on which a decision rule is trained to discriminate between examples belonging to *positive* and *negative* classes, where $\mathbf{x}_i$ represents a vector of d input variables describing the i^{th} example, and t_i indicates the class of the i^{th} example, where $t_i = 1$ if the example belongs to the positive class $\mathcal{C}_+$ and $t_i = 0$ if it belongs to the negative class $\mathcal{C}_-$. Kernel logistic regression aims to construct a familiar linear logistic regression model in a high-dimensional feature space induced by a Mercer kernel, giving rise to a non-linear form of logistic regression, i.e.

$$\operatorname{logit}\{y(\mathbf{x})\} = \mathbf{w} \cdot \boldsymbol{\phi}(\mathbf{x}) + b, \quad \text{where } \operatorname{logit}\{p\} = \log\left\{\frac{p}{1-p}\right\},$$

$\mathbf{w}$ is a vector of model parameters, $\boldsymbol{\phi}(\cdot)$ represents a non-linear transformation of the input vectors. Equivalently, we could write

$$y(\mathbf{x}) = \frac{1}{1 + \exp\{-\mathbf{w} \cdot \boldsymbol{\phi}(\mathbf{x}) - b\}}.$$

The logit *link* function constrains the output of the model to lie in the range $[0, \; 1]$. Rather than specifying the transformation $\boldsymbol{\phi} : \mathcal{X} \to \mathcal{F}$ directly, it is implied by a Mercer kernel, $\mathcal{K} : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$, which evaluates the inner product between the images of vectors in the *feature space*, $\mathcal{F}$,

$$\mathcal{K}(\mathbf{x}, \mathbf{x}') = \boldsymbol{\phi}(\mathbf{x}) \cdot \boldsymbol{\phi}(\mathbf{x}').$$

For the interpretation of the kernel function as an inner product in a fixed feature space to be valid, the kernel must obey Mercer's condition (Mercer 1909), i.e. the *kernel* or *Gram* matrix, $\mathbf{K} = [k_{ij} = \mathcal{K}(\mathbf{x}_i, \mathbf{x}_j)]_{i,j=1}^{\ell}$, must be positive (semi)-definite. The kernel most commonly used in practical applications of kernel learning methods is the squared exponential, or radial basis function (RBF), kernel,

$$\mathcal{K}(\mathbf{x}, \mathbf{x}') = \exp\{-\theta \|\mathbf{x} - \mathbf{x}'\|^2\}, \tag{1}$$

where θ is a kernel parameter controlling the sensitivity of the kernel. Interpreting the output of the kernel logistic regression model as an estimate of the *a-posteriori* probability of class membership, then provided the data represent an i.i.d. (independent and identically distributed) sample from a Bernoulli distribution conditioned on the input variables, the likelihood of the training data is given by

$$\mathcal{L} = \prod_{i=1}^{\ell} y_i^{t_i} [1 - y_i]^{1-t_i},$$

where $y_i = y(\mathbf{x}_i)$. The optimal vector of model parameters, $\mathbf{w}$, is found by minimizing a cost function representing the regularized (Tikhonov and Arsenin 1977) negative log-likelihood of the training data, in this case known as the *cross-entropy*,

$$E = \frac{1}{2}\|\mathbf{w}\|^2 - \frac{\gamma}{2}\sum_{i=1}^{\ell}[t_i \log\{y_i\} + (1 - t_i)\log\{1 - y_i\}], \tag{2}$$

where γ is a *regularization* parameter controlling the bias-variance trade-off (Geman et al. 1992). The *representer* theorem (Kimeldorf and Wahba 1971; Schölkopf et al. 2002) states that the solution to an optimization problem of this nature can be written in the form of a linear combination of the training patterns, i.e.

$$\mathbf{w} = \sum_{i=1}^{\ell} \alpha_i \boldsymbol{\phi}(\mathbf{x}_i),$$

which implies that

$$\operatorname{logit}\{y(\mathbf{x})\} = \sum_{i=1}^{\ell} \alpha_i \mathcal{K}(\mathbf{x}_i, \mathbf{x}) + b \quad \text{and} \quad \|\mathbf{w}\|^2 = \boldsymbol{\alpha}^\top \mathbf{K}\boldsymbol{\alpha},$$

where $\boldsymbol{\alpha} = (\alpha_1, \alpha_2, \ldots, \alpha_\ell)$ is a vector of *dual* model parameters. The benefit of the "kernel trick" then becomes apparent; it allows us to construct powerful linear models in very high (potentially infinite) dimensional feature spaces using mathematics involving only finite quantities, such as the $\ell \times \ell$ Gram matrix.

2.1 Iteratively re-weighted least squares training procedure

The objective function for a wide range of kernel learning methods, including kernel logistic regression, can be written in the form,

$$E = \frac{1}{2}\|\mathbf{w}\|^2 + \gamma \sum_{i=1}^{\ell} c(y_i, t_i) \tag{3}$$

where $c(\cdot, \cdot)$ is a convex loss function, in this case the negative log-likelihood assuming a Bernoulli trial, $c(y, t) = -[t \log y + (1 - t)\log(1 - y)]$. A closed form expression for the minimum of the objective function (3) is not immediately apparent, and so it is most easily minimized via an iteratively re-weighted least-squares (IRWLS) procedure, commonly used in training conventional logistic regression models and radial basis function (RBF) networks (Nabney 1999). Let z_i represent the output of the kernel machine for the i^{th} training pattern, prior to the non-linear transform,

$$z_i = \sum_{j=1}^{\ell} \alpha_j \mathcal{K}(\mathbf{x}_j, \mathbf{x}_i) + b.$$

In the case of kernel logistic regression, z_i represents the log-odds ratio. The first and second derivatives of the loss, with respect to z_i, are then given by

$$\frac{\partial c_i}{\partial z_i} = y_i - t_i \quad \text{and} \quad \frac{\partial^2 c_i}{\partial z_i^2} = y_i(1 - y_i),$$

where $c_i = c(y_i, t_i)$. As we are interested only in minimizing the convex loss function, we substitute a weighted least-squares criterion, providing a local approximation of c_i only up to some arbitrary constant, C, i.e.

$$q_i = \frac{\beta_i}{2}[\eta_i - z_i]^2 \approx c(y_i, t_i) + C.$$

Clearly, we require the curvature of q_i and c_i, with respect to z_i, to be identical at z_i, and therefore

$$\frac{\partial^2 q_i}{\partial z_i^2} = \frac{\partial^2 c_i}{\partial z_i^2} \implies \beta_i = y_i(1 - y_i).$$

We also require the gradient of q_i, with respect to z_i, to match that of c_i, such that

$$\frac{\partial q_i}{\partial z_i} = -\beta_i[\eta_i - z_i] = \frac{\partial c_i}{\partial z_i} \implies \eta_i = z_i - \frac{y_i - t_i}{y_i(1 - y_i)}.$$

The original objective function (2), can then be solved iteratively, alternating updates of the dual parameters, $(\boldsymbol{\alpha},\ b)$, via a regularized weighted least-squares loss function,

$$\widetilde{L}^{\sigma} = \frac{1}{2}\|\mathbf{w}\|^2 + \frac{\gamma}{2}\sum_{i=1}^{\ell} \beta_i[\eta_i - z_i]^2, \tag{4}$$

and updates of the weighting coefficients, $\boldsymbol{\beta} = (\beta_1, \beta_2, \ldots, \beta_\ell)$, and targets, $\boldsymbol{\eta} = (\eta_1, \eta_2, \ldots, \eta_\ell)$. In the case of kernel logistic regression, the update formulae are:

$$\beta_i = y_i(1 - y_i) \quad \text{and} \quad \eta_i = z_i - \frac{y_i - t_i}{y_i(1 - y_i)}. \tag{5}$$

The weighted least-squares problem (4) can also be solved via a system of linear equations, with a computational complexity of $\mathcal{O}(\ell^3)$ operations, as follows: Minimizing (4) can be recast in the form of a constrained optimization problem (Suykens et al. 2002a),

$$\min \mathcal{J} = \frac{1}{2}\|\mathbf{w}\|^2 + \frac{\gamma}{2}\sum_{i=1}^{\ell} \beta_i \varepsilon_i^2 \tag{6}$$

subject to

$$\eta_i = \mathbf{w} \cdot \boldsymbol{\phi}(\mathbf{x}_i) + b + \varepsilon_i, \quad \forall\, i \in \{1, 2, \ldots, \ell\}. \tag{7}$$

The primal Lagrangian for this optimization problem gives the unconstrained minimization problem,

$$\mathcal{L} = \frac{1}{2}\|\mathbf{w}\|^2 + \frac{\gamma}{2}\sum_{i=1}^{\ell} \beta_i \varepsilon_i^2 - \sum_{i=1}^{\ell} \alpha_i\{\mathbf{w} \cdot \boldsymbol{\phi}(\mathbf{x}_i) + b + \varepsilon_i - \eta_i\},$$

where $\boldsymbol{\alpha} = (\alpha_1, \alpha_2, \ldots, \alpha_\ell) \in \mathbb{R}^\ell$ is a vector of Lagrange multipliers. The optimality conditions for this problem can be expressed as follows:

$$\frac{\partial \mathcal{L}}{\partial \mathbf{w}} = \mathbf{0} \Longrightarrow \mathbf{w} = \sum_{i=1}^{\ell} \alpha_i \boldsymbol{\phi}(\mathbf{x}_i), \tag{8}$$

$$\frac{\partial \mathcal{L}}{\partial b} = 0 \Longrightarrow \sum_{i=1}^{\ell} \alpha_i = 0, \tag{9}$$

$$\frac{\partial \mathcal{L}}{\partial \varepsilon_i} = 0 \Longrightarrow \alpha_i = \beta_i \gamma \varepsilon_i, \quad \forall_i \in \{1, 2, \ldots, \ell\}, \tag{10}$$

$$\frac{\partial \mathcal{L}}{\partial \alpha_i} = 0 \Longrightarrow \mathbf{w} \cdot \boldsymbol{\phi}(\mathbf{x}_i) + b + \varepsilon_i - \eta_i = 0, \quad \forall\, i \in \{1, 2, \ldots, \ell\}. \tag{11}$$

Using (8) and (10) to eliminate $\mathbf{w}$ and $\boldsymbol{\varepsilon} = (\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_\ell)$, from (11), we find that

$$\sum_{j=1}^{\ell} \alpha_j \boldsymbol{\phi}(\mathbf{x}_j) \cdot \boldsymbol{\phi}(\mathbf{x}_i) + b + \frac{\alpha_i}{\gamma \beta_i} = \eta_i, \quad \forall\, i \in \{1, 2, \ldots, \ell\}. \tag{12}$$

Noting that $\mathcal{K}(\mathbf{x}, \mathbf{x}') = \boldsymbol{\phi}(\mathbf{x}) \cdot \boldsymbol{\phi}(\mathbf{x}')$, the system of linear equations can be written more concisely in matrix form as

$$\begin{bmatrix} \mathbf{K} + \frac{1}{\gamma}\mathbf{B} & \mathbf{1} \\ \mathbf{1}^\top & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{\alpha} \\ b \end{bmatrix} = \begin{bmatrix} \boldsymbol{\eta} \\ 0 \end{bmatrix}, \tag{13}$$

where $\mathbf{K} = [k_{ij} = \mathcal{K}(\mathbf{x}_i, \mathbf{x}_j)]_{i,j=1}^{\ell}$ and $\mathbf{B} = \operatorname{diag}\{\beta_1^{-1}, \beta_2^{-1}, \ldots, \beta_\ell^{-1}\}$. The optimal parameters for the kernel machine can then be obtained with a computational complexity of $\mathcal{O}(\ell^3)$ operations. Note that the iteratively re-weighted least-squares (IRWLS) procedure is equivalent to the application of Newton's method. We present the learning algorithm in terms of IRWLS here as the proposed approximate leave-one-out method is an extension of an existing exact method for weighted least-squares models.

2.2 Efficient implementation via Cholesky decomposition

A more efficient training algorithm can be obtained, taking advantage of the special structure of the system of linear equations. The system of linear equations (13) to be solved during each step of the iteratively re-weighted least squares procedure is given by,

$$\begin{bmatrix} \mathbf{M} & \mathbf{1} \\ \mathbf{1}^\top & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{\alpha} \\ b \end{bmatrix} = \begin{bmatrix} \boldsymbol{\eta} \\ 0 \end{bmatrix}, \tag{14}$$

where $\mathbf{M} = \mathbf{K} + \gamma^{-1}\mathbf{B}$. Unfortunately the matrix on the left-hand side is not positive definite, and so we cannot solve this system of linear equations directly using the Cholesky decomposition (Golub and Van Loan 1996). However, the first row of (14) can be re-written as

$$\mathbf{M}(\boldsymbol{\alpha} + \mathbf{M}^{-1}\mathbf{1}b) = \boldsymbol{\eta}. \tag{15}$$

Rearranging (15), we see that $\boldsymbol{\alpha} = \mathbf{M}^{-1}(\boldsymbol{\eta} - \mathbf{1}b)$, using this result to eliminate α, the second row of (14) can be written as,

$$\mathbf{1}^\top \mathbf{M}^{-1}\mathbf{1}b = \mathbf{1}^\top \mathbf{M}^{-1}\boldsymbol{\eta}. \tag{16}$$

The system of linear equations (14) can be then be solved by first solving two positive definite linear systems

$$\mathbf{M}\boldsymbol{\xi} = \mathbf{1} \quad \text{and} \quad \mathbf{M}\boldsymbol{\zeta} = \boldsymbol{\eta}, \tag{17}$$

and then updating the model parameters of the kernel logistic regression machine as follows:

$$b = \frac{\mathbf{1}^\top \boldsymbol{\zeta}}{\mathbf{1}^\top \boldsymbol{\xi}} \quad \text{and} \quad \boldsymbol{\alpha} = \boldsymbol{\zeta} - \boldsymbol{\xi} b.$$

The two systems of linear equations (17) can be solved efficiently using the Cholesky decomposition of $\mathbf{M} = \mathbf{R}^\top \mathbf{R}$, where $\mathbf{R}$ is the upper triangular Cholesky factor of $\mathbf{M}$ (Suykens et al. 2002b). Note that the computational complexity of the Cholesky decomposition is $\mathcal{O}(\ell^3)$, but that of the back-substitution used in solving the two systems of linear equations is only $\mathcal{O}(\ell^2)$ operations. As a result, the Cholesky decomposition is both computationally efficient as well as numerically more robust (Golub and Van Loan 1996).

3 Cross-validation based model selection strategies

The simplest form of model selection criterion typically partitions the available data into training, validation and test sets. The training set is used to determine the optimal values of the model parameters, an appropriate performance measure is evaluated over the validation or *hold-out* set in order to optimize the hyper-parameters and the test set is used to obtain an unbiased estimate of generalization performance. If data is relatively scarce, a cross-validation procedure is often used (Stone 1974). Under a k-fold cross-validation strategy, the data are partitioned into k subsets of approximately equal size. Models are then trained on each of the k combinations of $k-1$ subsets, in each case the performance of the model is estimated using the remaining subset not forming part of the training data for that model. The cross-validation estimate of a given performance metric is simply the mean of the performance in each fold of the cross-validation procedure. Cross-validation clearly makes better use of the available data as every pattern is used as both a training and a test pattern. The most extreme form of cross-validation, in which each subset contains only a single pattern, is known as leave-one-out cross-validation (Lachenbruch and Mickey 1968). Leave-one-out cross-validation is often used in model selection, partly as it is known to be approximately unbiased (Luntz and Brailovsky 1969), but also because it can be implemented very efficiently in the case of linear regression, least-squares kernel learning methods (Allen 1974; Cook and Weisberg 1982; Weisberg 1985; Cawley and Talbot 2003, 2004b; Bo et al. 2006; Rasmussen and Williams 2006) and k-nearest neighbor methods, or approximated efficiently in the case of the support vector machine (Chapelle et al. 2002). In this paper, in addition to investigating conventional cross-validation based model selection strategies, we also propose an approximate leave-one-out cross-validation method for kernel logistic regression based on existing efficient methods for least-squares kernel learning methods, the method presented here being a greatly refined version of the method briefly outlined in Cawley and Talbot (2004a). However, the approach is quite general, and can easily be applied to any kernel regression method with a convex loss function, $c(\cdot, \cdot)$.

3.1 Approximate leave-one-out cross-validation

The optimal values for the parameters of a kernel regression model are iteratively determined via a sequence of weighted least-squares optimization problems. It is well known that leave-one-out cross-validation of least-squares models can be performed very efficiently in closed form (Cook and Weisberg 1982; Weisberg 1985; Green and Silverman 1994; Cawley and Talbot 2003; Cawley and Talbot 2004b). These methods can be extended to provide an approximate leave-one-out cross-validation method for kernel regression methods with an arbitrary loss, via exact leave-one-out cross-validation of the quadratic approximation (4) of the true loss minimized in the final iteration of the IRWLS training procedure (Green and Silverman 1994; Cawley and Talbot 2004a). The matrix on the left-hand side of (13) can be decomposed into block-matrix representation, as follows:

$$\begin{bmatrix} \mathbf{K}+\gamma^{-1}\mathbf{B} & \mathbf{1} \\ \mathbf{1}^\top & 0 \end{bmatrix} = \begin{bmatrix} c_{11} & \mathbf{c}_1^\top \\ \mathbf{c}_1 & \mathbf{C}_1 \end{bmatrix} = \mathbf{C}. \tag{18}$$

Let $[\boldsymbol{\alpha}^{(-i)}; b^{(-i)}]$ represent the parameters of the kernel machine during the i^{th} iteration of the leave-one-out cross-validation procedure, then in the first iteration, in which the first training pattern is excluded,

$$\begin{bmatrix} \boldsymbol{\alpha}^{(-1)} \\ b^{(-1)} \end{bmatrix} = \mathbf{C}_1^{-1}[\eta_2, \ldots, \eta_\ell, 0]^\top.$$

The leave-one-out prediction for the first training pattern is then given by,

$$\hat{z}_1^{(-1)} = \mathbf{c}_1^\top \begin{bmatrix} \boldsymbol{\alpha}^{(-1)} \\ b^{(-1)} \end{bmatrix} = \mathbf{c}_1^\top \mathbf{C}_1^{-1}[\eta_2, \ldots, \eta_\ell, 0]^\top.$$

Considering the last ℓ equations in the system of linear equations (13), it is clear that $[\mathbf{c}_1\ \mathbf{C}_1][\alpha_1, \ldots, \alpha_\ell, b]^\top = [\eta_2, \ldots, \eta_\ell, 0]^\top$, and so

$$\hat{z}_1^{(-1)} = \mathbf{c}_1^\top \mathbf{C}_1^{-1}[\mathbf{c}_1\ \mathbf{C}_1][\boldsymbol{\alpha}^\top, b]^\top = \mathbf{c}_1^\top \mathbf{C}_1^{-1}\mathbf{c}_1\alpha_1 + \mathbf{c}_1\,[\alpha_2, \ldots, \alpha_\ell, b]^\top.$$

Noting, from the first equation in the system of linear equations (13), that $\eta_1 = c_{11}\alpha_1 + \mathbf{c}_1^\top[\alpha_2, \ldots, \alpha_\ell, b]^\top$, thus

$$\hat{z}_1^{(-1)} = \eta_1 - \alpha_1(c_{11} - \mathbf{c}_1^\top \mathbf{C}_1^{-1}\mathbf{c}_1).$$

Finally, via the block matrix inversion lemma,

$$\begin{bmatrix} c_{11} & \mathbf{c}_1^\top \\ \mathbf{c}_1 & \mathbf{C}_1 \end{bmatrix}^{-1} = \begin{bmatrix} \kappa^{-1} & -\kappa^{-1}\mathbf{c}_1\mathbf{C}_1^{-1} \\ \mathbf{C}_1^{-1} + \kappa^{-1}\mathbf{C}_1^{-1}\mathbf{c}_1^\top\mathbf{c}_1\mathbf{C}_1^{-1} & -\kappa^{-1}\mathbf{C}_1^{-1}\mathbf{c}_1^\top \end{bmatrix}, \tag{19}$$

where $\kappa = c_{11} - \mathbf{c}_1^\top \mathbf{C}_1^{-1}\mathbf{c}$, and noting that the system of linear equations (13) is insensitive to permutations of the ordering of the equations and of the unknowns, we have that,

$$\hat{z}_i^{(-i)} = \eta_i - \frac{\alpha_i}{\mathbf{C}_{ii}^{-1}}. \tag{20}$$

This means that, assuming the system of linear equations (13) is solved via explicit inversion of $\mathbf{C}$, an approximate leave-one-out cross-validation estimate of the test loss (22) can be evaluated using information already available as a by-product of training the least-squares support vector machine on the entire dataset. This approximation is based on the assumption that the parameters of the quadratic approximation of the regularized loss function, $\boldsymbol{\beta}$ and $\boldsymbol{\eta}$ are essentially unchanged during the leave-one-out cross-validation procedure (c.f. Green and Silverman 1994). Essentially we substitute a leave-one-out cross-validation of the model using the quadratic approximation for a leave-one-out cross-validation using the true loss. Alternatively, as described in Appendix A, we can view the approximation as taking a single Newton step of the reduced training criterion in each fold of the leave-one-out procedure, starting from the vector of model parameters minimizing the regularized loss on the entire training sample.

3.2 Efficient implementation via Cholesky factorization

The approximate leave-one-out cross-validation estimator for kernel logistic regression is described by (20). The coefficients of the kernel expansion, $\boldsymbol{\alpha}$, can be found efficiently, via iteratively re-weighted least squares based on Cholesky factorization, as described in Sect. 2.2. However we must also determine the diagonal elements of $\mathbf{C}^{-1}$ in an efficient manner. Using the block matrix inversion formula, we obtain

$$\mathbf{C}^{-1} = \begin{bmatrix} \mathbf{M} & \mathbf{1} \\ \mathbf{1}^\top & 0 \end{bmatrix}^{-1} = \begin{bmatrix} \mathbf{M}^{-1} + \mathbf{M}^{-1}\mathbf{1}\mathbf{S}_M^{-1}\mathbf{1}^\top\mathbf{M}^{-1} & -\mathbf{M}^{-1}\mathbf{1}\mathbf{S}_M^{-1} \\ -\mathbf{S}_M^{-1}\mathbf{1}^\top\mathbf{M}^{-1} & \mathbf{S}_M^{-1} \end{bmatrix}$$

where $\mathbf{M} = \mathbf{K} + \gamma^{-1}\mathbf{B}$ and $\mathbf{S}_M = -\mathbf{1}^\top\mathbf{M}^{-1}\mathbf{1} = -\mathbf{1}^\top\boldsymbol{\xi}$ is the Schur complement of $\mathbf{M}$. The inverse of the positive definite matrix, $\mathbf{M}$, can be computed efficiently from its Cholesky factorization, via the `SYMINV` algorithm (Seaks 1972), for example using the LAPACK (Anderson et al. 1999) routine `DTRTRI`. Let $\mathbf{R} = [r_{ij}]_{i,j=1}^n$ be the lower triangular Cholesky factor of the positive definite matrix $\mathbf{M}$, such that $\mathbf{M} = \mathbf{R}\mathbf{R}^\top$. Furthermore, let

$$\mathbf{S} = [s_{ij}]_{i,j=1}^n = \mathbf{R}^{-1}, \quad \text{where } s_{ii} = \frac{1}{r_{ii}} \text{ and } s_{ij} = -s_{ii}\sum_{k=1}^{i-1} r_{ik}s_{kj},$$

represent the (lower triangular) inverse of the Cholesky factor. The inverse of $\mathbf{M}$ is then given by $\mathbf{M}^{-1} = \mathbf{S}^\top\mathbf{S}$. In the case of efficient approximate leave-one-out cross-validation of kernel logistic regression machines, we are principally concerned only with the diagonal elements of $\mathbf{M}^{-1}$, given by

$$M_{ii}^{-1} = \sum_{j=1}^{i} s_{ij}^2 \implies C_{ii}^{-1} = \sum_{j=1}^{i} s_{ij}^2 + \frac{\xi_i^2}{S_M}, \quad \forall\, i \in \{1, 2, \dots, \ell\}.$$

The computational complexity of the basic training algorithm is $\mathcal{O}(\ell^3)$ operations, being dominated by the evaluation of the Cholesky factor. However, the computational complexity of the analytic approximate leave-one-out cross-validation approximation, when performed as a by-product of the training algorithm, is only $\mathcal{O}(\ell)$ operations. The computational expense of the leave-one-out cross-validation procedure therefore rapidly becomes negligible as the training set becomes larger.

Note that the proposed approximate leave-one-out cross-validation procedure is quite general, and can be applied to kernel regression machines with an essentially arbitrary convex loss, for instance kernel Poisson regression (Cawley et al. 2007). The support vector machine (Boser et al. 1992; Cortes and Vapnik 1995) can also be trained in primal form via Newton's method (Chapelle 2007), where the approximate leave-one-out cross-validation method given here is equivalent to the span bound (Vapnik and Chapelle 2000; Chapelle et al. 2002). It should be noted, however, that the proposed leave-one-out cross-validation method is not suitable for large scale applications in its current form, due to the computational complexity of $\mathcal{O}(\ell^3)$ operations. For large scale applications, sparse algorithms, such as the import vector machine (Zhu and Hastie 2005), could be used. However, if a sparse set of basis vectors has been identified, an approximate leave-one-out cross-validation method for sparse kernel logistic regression would be feasible based on the corresponding approach for sparse least-squares kernel machines (Cawley and Talbot 2004a, 2004b).

3.3 Optimization strategies

In practical applications of kernel learning methods the hyper-parameters are most often selected via a simple grid-based search method, in which the model selection criterion is evaluated at a set of points, forming a regular grid with even spacing, normally over a logarithmic scale. An improved hierarchical grid search procedure repeats this process, each time using a refined grid centered on the best solution found at the previous scale. However, grid-search procedures rapidly become computationally unfeasible as the number of hyper-parameters to be optimized grows larger than only two or three. However, if the model selection strategy is a relatively smooth function of the hyper-parameters, the Nelder-Mead simplex optimization algorithm (Nelder and Mead 1965), as implemented by the `fminsearch` routine of the MATLAB optimization toolbox, provides a simple and efficient alternative, as long as the number of hyper-parameters remains relatively small (less than about ten). For problems with a larger vector of hyper-parameters, gradient-based methods are likely to be more efficient, for instance conjugate-gradient methods, as implemented by the `fminunc` routine of the MATLAB optimization toolbox, or scaled conjugate gradient optimization (Williams 1991). Appendix B provides the derivation of gradient information with a computational complexity of $\mathcal{O}(\ell^3 + d\ell^2)$ operations, where d is the number of kernel parameters. Note that for the Gaussian process classifier based on the Laplace approximation, the gradient of the marginal likelihood, with respect to the parameters of the covariance function, can be evaluated with a complexity of $\mathcal{O}(d\ell^2)$ operations (provided the inverse of the covariance matrix is available as a by-product of fitting the model). The computational expense of model selection for LOO-KLR and L-GPC can thus be expected to exhibit similar scaling. However, when fitting models with many hyper-parameters there is a danger of over-fitting the model selection criterion, and so the addition of a regularization term to the selection criterion may be beneficial (Cawley and Talbot 2007). It should be noted however, that the use of gradient-based methods does not necessitate the analytic computation of gradient information (c.f. Bo et al. 2006) as the required partial derivatives can also be approximated by the method of finite-differences, albeit with an increased complexity of $\mathcal{O}(d\ell^3)$ operations.

3.4 Model selection for Gaussian process classifiers

An estimate of the leave-one-out cross-validation loss is also available as a by-product of fitting a Gaussian process classifier using the Expectation Propagation algorithm (Minka 2001; Rasmussen and Williams 2006). This approach is equivalent to the mean-field methods introduced by Opper and Winther (2000). Sundararajan and Keerthi (2001) present a leave-one-out cross-validation procedure for hyper-parameter selection in Gaussian process regression, also discussed in Rasmussen and Williams (2006). The Gaussian process classifier based on the Laplace approximation (Williams and Barber 1998) also uses a model selection criterion, in this case the marginal likelihood, evaluated using a quadratic approximation to the regularized loss function. The model selection process for Gaussian process classifiers under the Laplace approximation thus bears some similarities with the proposed approximate leave-one-out cross-validation method.

4 Results

We begin by investigating the accuracy of the approximate leave-one-out method using Ripley's `synthetic` data. Figure 1 shows a kernel logistic regression model of this dataset, using an isotropic radial basis function kernel. Figure 2 shows contour plots of the cross-entropy loss for a kernel logistic regression model of Ripley's `synthetic` data, using an isotropic radial basis function kernel, computed using the proposed approximate leave-one-out method, exact leave-one-out, 10-fold cross-validation and the test loss as a function of the hyper-parameters. It is clear that the approximate leave-one-out loss behaves in a similar manner to the exact leave-one-out loss, without being exactly identical. Most importantly, the minimum of all three estimates of the true loss are in accordance with the loss computed over the independent test set. Note that the loss is a smooth function of the hyper-parameters and so is well suited to automated model selection using standard non-linear optimization methods. Figure 3 shows the exact and approximate leave-one-out cross-entropy loss as a function of each of the hyper-parameters, holding the other constant at its optimal value. The approximate leave-one-out estimator is clearly of sufficient accuracy for model selection purposes, but would not be suitable for performance evaluation, except for well-tuned models. Note also that cross-validation based model selection criteria are not necessarily unimodal.

Next, we present experimental results demonstrating the accuracy and efficiency of the proposed approximate leave-one-out cross-validation model selection procedure for kernel logistic regression. Table 1 shows a comparison of the error rates of kernel logistic regression, using the proposed approximate leave-one-out cross-validation based model selection process, and a variety of other state-of-the-art statistical pattern recognition algorithms over the suite of thirteen public domain benchmark datasets used in the study by Mika et al.

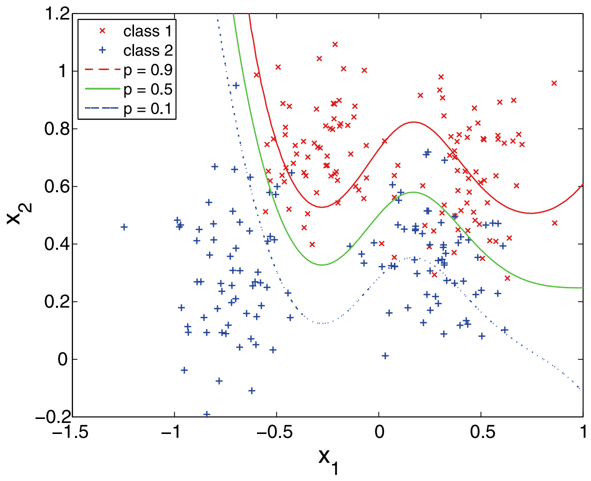

Fig. 1 Kernel logistic regression model of Ripley's `synthetic` data, with isotropic radial basis function kernel and approximate leave-one-out cross-validation based model selection

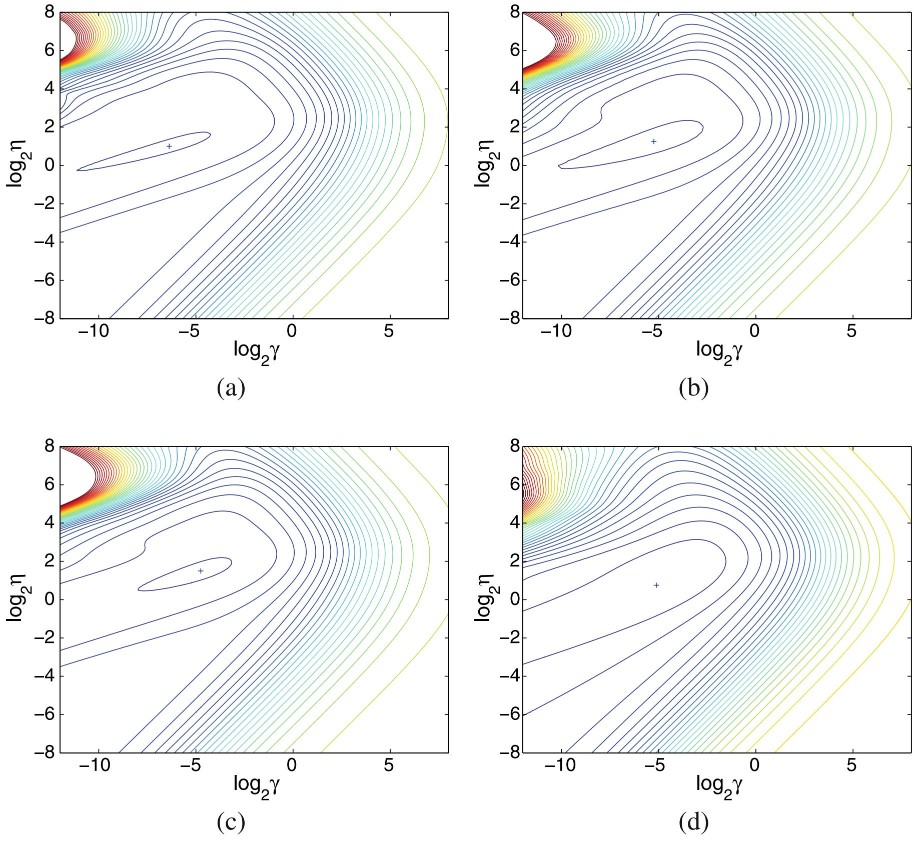

Fig. 2 Contour plots of (**a**) the approximate leave-one-out cross-entropy loss, (**b**) the exact leave-one-out loss, (**c**) the 10-fold cross-validation loss and (**d**) the test set loss for the `synthetic` benchmark, for a kernel logistic regression model, as a function of the regularization parameter, γ, and the kernel parameter, η. The minimum loss is indicated by a *cross*, $+$

(2000). The same set of 100 random partitions of the data (20 in the case of the `image` and `splice` benchmarks) to form training and test sets used in that study are also used here. In the case of the LOO-KLR, KLR, EP-GPC and LOO-KFD algorithms, model selection is performed independently for each realization of the dataset, such that the standard errors reflect the variability of both the training algorithm and the model selection procedure with changes in the sampling of the data. For the LOO-KLR and KLR methods, model selection was performed via the Nelder-Mead simplex algorithm (Nelder and Mead 1965). The isotropic squared exponential (RBF) kernel is used for all kernel learning methods, including the L-GPC and EP-GPC, chosen due to their state-of-the-art performance and similar structure.

Table 2 shows the model selection time for leave-one-out and 10-fold cross-validation based kernel logistic regression (LOO-KLR and KLR respectively) and expectation propagation and Laplace approximation based Gaussian process classifiers (EP-GPC and L-GPC). Table 3 shows the average amount of information about the test set labels in excess of pre-

Table 1 Error rates of various state-of-the-art classifiers over thirteen benchmark datasets, (LOO-KLR) kernel logistic regression using the proposed leave-one-out cross-validation model selection procedure, (KLR) kernel logistic regression with conventional 10-fold cross-validation based model selection, Gaussian process classifier implemented via the Laplace approximation and expectation propagation algorithm (L-GPC and EP-GPC respectively), (LOO-KFD) kernel Fisher discriminant with leave-one-out cross-validation based model selection procedures (Cawley and Talbot 2003), (SVM) support vector machine (Boser et al. 1992; Cortes and Vapnik 1995) and (KFD) kernel Fisher discriminant classifier (Mika et al. 1999). The results for models SVM and KFD are taken from the study by Mika et al. (2000). The results for model LOO-KFD are taken from Cawley and Talbot (2003). The results for the L-GPC and EP-GPC were obtained using the GPML MATLAB toolbox accompanying the book by Rasmussen and Williams (2006). The results for each method are presented in the form of the mean error rate over test data for 100 realizations of each dataset (20 in the case of the image and splice datasets), along with the associated standard error. The best results are shown in boldface and the second best in italics (without implication of statistical significance)

Dataset	LOO-KLR	KLR	L-GPC	EP-GPC	LOO-KFD	SVM	KFD
Banana	10.6 ± 0.05	*10.5 ± 0.05*	**10.4 ± 0.05**	**10.4 ± 0.05**	**10.4 ± 0.04**	11.5 ± 0.07	10.8 ± 0.05
Breast cancer	26.6 ± 0.47	26.5 ± 0.49	26.5 ± 0.48	26.5 ± 0.49	26.3 ± 0.42	*26.0* ± 0.47	**25.8 ± 0.46**
Diabetes	23.4 ± 0.18	23.5 ± 0.16	23.3 ± 0.18	23.3 ± 0.18	**23.1 ± 0.18**	23.5 ± 0.17	*23.2 ± 0.16*
Flare solar	34.3 ± 0.17	34.2 ± 0.16	34.2 ± 0.18	34.2 ± 0.21	34.2 ± 1.16	**32.4 ± 0.18**	*33.2 ± 0.17*
German	*23.5 ± 0.21*	*23.5 ± 0.21*	**23.4 ± 0.21**	**23.4 ± 0.21**	23.6 ± 0.20	23.6 ± 0.21	23.7 ± 0.22
Heart	16.6 ± 0.31	16.6 ± 0.31	16.4 ± 0.28	16.7 ± 0.29	**15.9 ± 0.35**	*16.0 ± 0.33*	16.1 ± 0.34
Image	3.1 ± 0.13	3.1 ± 0.11	**2.8 ± 0.10**	**2.8 ± 0.12**	4.0 ± 0.06	*3.0 ± 0.06*	3.3 ± 0.06
Ringnorm	1.6 ± 0.02	1.6 ± 0.01	5.9 ± 0.08	4.4 ± 0.06	**1.4 ± 0.08**	1.7 ± 0.01	**1.5** ± *0.01*
Splice	11.2 ± 0.18	11.2 ± 0.17	12.3 ± 0.18	11.6 ± 0.18	*10.8 ± 0.07*	10.9 ± 0.07	**10.5 ± 0.06**
Thyroid	*4.3 ± 0.21*	4.7 ± 0.21	4.5 ± 0.21	4.4 ± 0.22	4.5 ± 0.20	4.8 ± 0.22	**4.2 ± 0.21**
Titanic	22.6 ± 0.10	22.6 ± 0.09	22.6 ± 0.13	22.6 ± 0.13	**22.3 ± 0.12**	*22.4 ± 0.10*	23.2 ± 0.20
Twonorm	2.9 ± 0.03	2.9 ± 0.03	3.1 ± 0.04	3.1 ± 0.03	*2.7 ± 0.02*	3.0 ± 0.02	**2.6 ± 0.02**
Waveform	*9.9 ± 0.04*	*9.9 ± 0.04*	10.0 ± 0.04	10.1 ± 0.05	**9.7 ± 0.04**	*9.9 ± 0.04*	*9.9 ± 0.04*

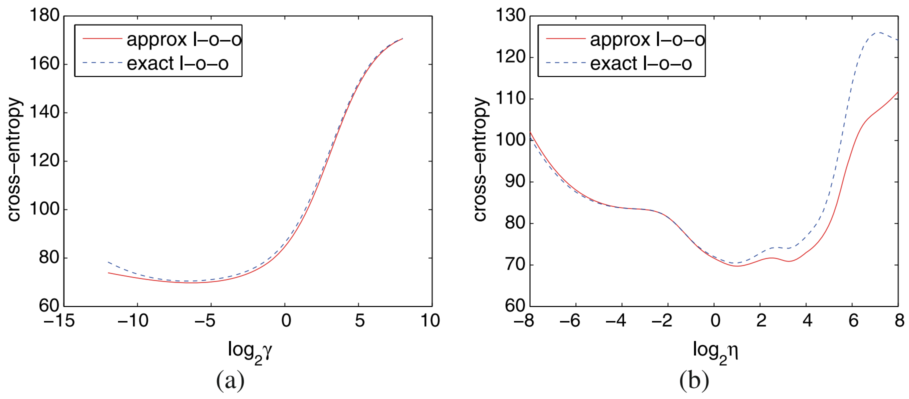

Fig. 3 Plot of the approximate and exact leave-one-out cross-entropy loss, as a function of (**a**) the regularization parameter, γ, with the kernel parameter, η, held at its optimal value and (**b**) the kernel parameter, with the regularization parameter held at its optimal value

Table 2 Model selection time for kernel logistic regression models with leave-one-out and 10-fold cross-validation based model selection (LOO-KLR and KLR respectively) and Gaussian process classifiers based on the Laplace approximation and expectation propagation (L-GPC and EP-GPC respectively) over thirteen benchmark datasets

Dataset	Selection time (seconds)			
	LOO-KLR	KLR	L-GPC	EP-GPC
Banana	**37.5 ± 1.231**	**199.1 ± 7.995**	*44.6 ± 0.587*	916.9 ± 20.51
Breast cancer	**5.4 ± 0.210**	28.7 ± 1.030	*16.0* ± 0.269	*126.2* ± *1.919*
Diabetes	**37.9 ± 0.939**	201.9 ± 5.803	*111.5* ± 1.746	1129.2 ± 22.10
Flare solar	**90.1 ± 4.703**	429.6 ± 22.33	*273.7 ± 3.610*	3108.9 ± 75.08
German	**99.5 ± 2.687**	*531.8 ± 12.03*	566.9 ± 6.853	3760.4 ± 67.04
Heart	**3.1 ± 0.092**	*18.2 ± 0.645*	18.5 ± 0.227	116.6 ± 2.177
Image	**832.6 ± 94.82**	4537.5 ± 168.0	*3643.6 ± 378.3*	25764.9 ± *2314*
Ringnorm	**60.9 ± 1.948**	359.9 ± 6.555	*216.8 ± 3.825*	1544.7 ± 43.35
Splice	**237.4 ± 10.84**	*1325.7 ± 74.08*	4054.2 ± 122.72	12653.7 ± 1480
Thyroid	**5.2 ± 0.348**	21.6 ± 0.839	*9.0 ± 0.278*	133.4 ± 4.049
Titanic	**3.3 ± 0.192**	11.5 ± 0.447	*5.1 ± 0.053*	83.8 ± 2.669
Twonorm	**46.7 ± 1.218**	270.0 ± 7.240	*245.6 ± 3.722*	1302.0 ± 45.05
Waveform	**46.5 ± 0.998**	260.2 ± 6.509	*222.0 ± 2.263*	1145.2 ± 35.38

dictions based on the prior class frequencies,

$$I = 1 + \frac{1}{n}\sum_{i=1}^{n} t_i \log_2 y_i + (1 - t_i)\log_2(1 - y_i)$$

where n is the number of test patterns. This statistic provides a measure of the accuracy of the predictions of *a-posteriori* probability obtained from a model (note that it is closely related to the cross-entropy).

Table 3 Mean target information for kernel logistic regression models with leave-one-out and 10-fold cross-validation based model selection (LOO-KLR and KLR respectively) and Gaussian process classifier based on the Laplace approximation and expectation propagation (L-GPC and EP-GPC respectively) over thirteen benchmark datasets

Dataset	Information (bits)			
	LOO-KLR	KLR	L-GPC	EP-GPC
Banana	0.648 ± 0.002	*0.653 ± 0.001*	0.647 ± 0.001	**0.655 ± 0.001**
Breast cancer	**0.222 ± 0.007**	**0.223 ± 0.007**	**0.228 ± 0.006**	*0.227 ± 0.007*
Diabetes	0.306 ± 0.003	0.306 ± 0.003	**0.312 ± 0.003**	*0.311 ± 0.003*
Flare solar	*0.172 ± 0.002*	*0.172 ± 0.002*	**0.174 ± 0.002**	**0.174 ± 0.002**
German	0.294 ± 0.004	0.296 ± 0.004	*0.297 ± 0.004*	**0.298 ± 0.004**
Heart	0.421 ± 0.007	*0.424 ± 0.007*	**0.425 ± 0.007**	0.421 ± 0.008
Image	*0.876 ± 0.005*	0.875 ± 0.005	0.671 ± 0.010	**0.881 ± 0.004**
Ringnorm	*0.930 ± 0.002*	**0.932 ± 0.001**	0.613 ± 0.001	0.757 ± 0.003
Splice	**0.613 ± 0.004**	**0.613 ± 0.004**	0.532 ± 0.001	*0.589 ± 0.003*
Thyroid	0.806 ± 0.023	*0.837 ± 0.012*	0.708 ± 0.012	**0.858 ± 0.006**
Titanic	0.182 ± 0.011	0.242 ± 0.004	**0.257 ± 0.002**	**0.256 ± 0.002**
Twonorm	*0.885 ± 0.001*	**0.886 ± 0.001**	0.872 ± 0.001	0.880 ± 0.001
Waveform	**0.675 ± 0.001**	**0.675 ± 0.001**	0.654 ± 0.001	*0.668 ± 0.001*

The use of multiple training/test partitions allows an estimate of the statistical significance of differences in performance between algorithms to be computed. Let $\hat{x}$ and $\hat{y}$ represent the means of the performance statistic for a pair of competing algorithms, and e_x and e_y the corresponding standard errors, then the z statistic is computed as

$$z = \frac{\hat{y} - \hat{x}}{\sqrt{e_x^2 + e_y^2}}.$$

The z-score can then be converted to a significance level via the normal cumulative distribution function, such that $z = 1.64$ corresponds to a 95% significance level. All statements of statistical significance in the remainder of this section refer to a 95% level of significance. Comparison of leave-one-out and 10-fold cross-validation based model selection strategies for kernel logistic regression reveals that the performance of both model selection strategies are very similar in terms of mean error rate. None of the differences in mean error rate for the two algorithms, shown in Table 1, are statistically significant at the 95% level. The estimates of *a-posteriori* probability obtained using these approaches are generally very similar, with k-fold cross-validation being statistically superior on only two benchmarks (`BANANA` and `TITANIC`). The kernel Fisher discriminant classifier appears to perform better than KLR or GPC models in terms of error rate, however it should be noted that the KFD classifier does not attempt to estimate the conditional probability of class membership, and so has an easier learning task, that concentrates more strongly on the decision boundary. Table 2 also shows that the approximate leave-one-out cross-validation based model selection is significantly less expensive, being approximately five times faster than the 10-fold cross-validation based approach, even though the latter had been extensively optimized (e.g. to prevent redundant evaluation of the kernel matrix). The computational complexity of the L-GPC, EP-GPC, KLR and LOO-KLR procedures are all $\mathcal{O}(\ell^3)$ operations (being dominated by the cost of fitting the initial model), so the model selection times essentially differ only by a constant factor for a given dataset.

The Gaussian process classifier, based on the expectation propagation algorithm (EP-GPC) with hyper-parameters selected so as to maximize the marginal likelihood, represents *the* state-of-the-art model having the same basic structure and design goals as kernel logistic regression (see e.g. Rasmussen and Williams 2006). The EP-GPC therefore provides a stern test of the proposed model selection algorithm. The Gaussian process classifier based on the Laplace approximation (L-GPC) is also relevant; like the LOO-KLR, it also adopts a model selection criterion, in this case the marginal likelihood, based on a quadratic approximation of the loss. The L-GPC provides very similar results, but with much lower computational expense. Table 1 shows the performance of LOO-KLR and EP-GPC to be generally comparable, in terms of error rate, with neither method dominating over all benchmarks. LOO-KLR is *statistically* superior to the EP-GPC on three benchmark datasets (RINGNORM, TWONORM and WAVEFORM) and *statistically* inferior on two (BANANA and IMAGE). In terms of predictive information (Table 3) again neither method uniformly dominates, with EP-GPC being *statistically* superior on three benchmarks (BANANA, THYROID and TITANIC) and *statistically* inferior on four (RINGNORM, SPLICE, TWONORM and WAVEFORM). This is a surprising result as the EP-GPC moderates the output by marginalizing over the posterior distribution of the model parameters, and therefore might be expected to produce more accurate estimates of the *a-posteriori* probability of class membership. However, the model selection criterion for the GPC models gives the probability of the data, *given the assumptions of the model* (Rasmussen and Williams 2006). Cross-validation based approaches, on the other hand, provide an estimate of generalization performance that does not depend on the model assumptions, and so may be more robust against model mis-specification (Wahba 1990). The results suggests the performance of the LOO-KLR algorithm is at least on a par with the EP-GPC in terms of generalization performance, whilst being typically 20 times faster. Note that the LOO-KLR model is also consistently faster than the L-GPC.

5 Conclusions

Model selection is an important step in practical applications of kernel learning methods, and must be performed in a diligent manner in order to obtain near-optimal generalization performance. In this paper we have proposed a close approximation of the leave-one-out cross-validation procedure for kernel logistic regression, which can be performed efficiently in closed form, providing a convenient means for automated model selection. An extensive experimental comparison has shown this method to be competitive in terms of performance with conventional k-fold cross-validation based model selection and with state-of-the-art Bayesian model selection principles, embodied by the expectation propagation based Gaussian process classifier. The proposed model selection technique is also demonstrated to be significantly faster than either of the alternative approaches investigated. The approach can easily be adapted to form efficient model selection procedures for a wide range of other kernel learning methods, for instance for use in survival analysis (e.g. Cawley et al. 2006). A public domain MATLAB implementation of the efficient approximate leave-one-out procedure described in this paper is available from http://theoval.cmp.uea.ac.uk/~gcc/projects/gkm/.

Acknowledgements We thank Olivier Chapelle and anonymous reviewers for their careful and constructive comments that have greatly improved this paper. We are particularly indebted to Olivier for providing the clever trick (26) for evaluating gradient information more efficiently and (with one of the reviewers) for pointing out the alternate derivation of the approximation, described in Appendix A.

Appendix A: Alternate derivation of the approximation leave-one-out criterion

Adapting the approach of Chapelle (2006), we consider a generalized kernel machine without an unregularized bias parameter, such that

$$f\{y_i\} = z_i = \sum_{j=1}^{\ell} \tilde{\alpha}_j \mathcal{K}(\mathbf{x}_j, \mathbf{x}_i),$$

where the vector of dual model parameters, $\tilde{\boldsymbol{\alpha}}$, minimizes the regularized training criterion (3) over the full set of training samples. In each iteration of the leave-one-out procedure (for notational convenience we consider the ℓ^{th}), we must find a set of model parameters, $\hat{\boldsymbol{\alpha}}$, minimizing a reduced training criterion,

$$\hat{E} = \frac{1}{2}\boldsymbol{\alpha}^\top \mathbf{K}\boldsymbol{\alpha} + \gamma \sum_{i=1}^{\ell-1} c(y_i, t_i).$$

However, a useful approximation can be found by taking only a single Newton step of the reduced criterion, $\hat{E}$, starting from the optimal parameters for the full training set, $\tilde{\alpha}$, without iterating to convergence, i.e.

$$\hat{\boldsymbol{\alpha}} \approx \bar{\boldsymbol{\alpha}} = \tilde{\boldsymbol{\alpha}} - \hat{\mathbf{H}}^{-1}\hat{\boldsymbol{\nabla}},$$

where $\mathbf{H}$ and $\boldsymbol{\nabla}$ represent the Hessian matrix and gradient vector of the reduced training criterion evaluated at $\tilde{\boldsymbol{\alpha}}$, given by

$$\hat{\mathbf{H}} = \mathbf{K} + \gamma \mathbf{K}\hat{\mathbf{D}}\mathbf{K} \quad \text{where } \hat{\mathbf{D}} = \text{diag}\left\{\frac{\partial^2 c_1}{\partial z_1^2}, \frac{\partial^2 c_2}{\partial z_2^2}, \ldots, \frac{\partial c_{\ell-1}^2}{\partial z_{\ell-1}^2}, 0\right\}$$

and

$$\hat{\boldsymbol{\nabla}} = \mathbf{K}(\tilde{\boldsymbol{\alpha}} + \gamma\hat{\mathbf{g}}) \quad \text{where } \hat{\mathbf{g}} = \left(\frac{\partial c_1}{\partial z_1}, \frac{\partial c_2}{\partial z_2}, \ldots, \frac{\partial c_{\ell-1}}{\partial z_{\ell-1}}, 0\right).$$

However, as $\tilde{\boldsymbol{\alpha}}$ is the minimizer of the original training criterion, we know that

$$\left.\frac{\partial E}{\partial \boldsymbol{\alpha}}\right|_{\boldsymbol{\alpha}=\tilde{\boldsymbol{\alpha}}} = \mathbf{K}(\tilde{\boldsymbol{\alpha}} + \gamma\mathbf{g}) = \mathbf{0} \quad \text{where } \mathbf{g} = \left(\frac{\partial c_i}{\partial z_i}\right)_{i=1}^{\ell},$$

and thus $\tilde{\alpha}_i + \gamma g_i = 0 = \tilde{\alpha}_i + \gamma \hat{g}_i, \ \forall\, i \neq \ell$, such that

$$\hat{\boldsymbol{\nabla}} = \mathbf{K}\begin{pmatrix}\mathbf{0} \\ \tilde{\alpha}_\ell\end{pmatrix}.$$

Noting that $\hat{D}_{\ell\ell} = 0$, the Newton step is then given by

$$\begin{aligned}\bar{\boldsymbol{\alpha}} - \tilde{\boldsymbol{\alpha}} = -\hat{\mathbf{H}}^{-1}\hat{\boldsymbol{\nabla}} &= -[\mathbf{K} + \gamma\mathbf{K}\hat{\mathbf{D}}\mathbf{K}]^{-1}\mathbf{K}\begin{pmatrix}\mathbf{0} \\ \tilde{\alpha}_\ell\end{pmatrix} \\ &= -\begin{bmatrix}\mathbf{I} + \gamma\hat{\mathbf{D}}_1\mathbf{K}_1 & \gamma\hat{\mathbf{D}}_1\mathbf{k} \\ \mathbf{0}^\top & 1\end{bmatrix}^{-1}\begin{pmatrix}\mathbf{0} \\ \tilde{\alpha}_\ell\end{pmatrix},\end{aligned}$$

where

$$\mathbf{K} = \begin{bmatrix} \mathbf{K}_1 & \mathbf{k} \\ \mathbf{k}^\top & K_{\ell\ell} \end{bmatrix}, \qquad \hat{\mathbf{D}} = \begin{bmatrix} \hat{\mathbf{D}}_1 & \mathbf{0} \\ \mathbf{0}^\top & \hat{D}_{\ell\ell} \end{bmatrix}, \qquad \mathbf{B}_1 = \hat{\mathbf{D}}_1^{-1}.$$

Using block matrix inversion lemma (19) we can re-write the Newton step as follows,

$$\begin{aligned} \hat{\mathbf{H}}^{-1}\hat{\nabla} &= \begin{bmatrix} [\mathbf{I} + \gamma \hat{\mathbf{D}}_1 \mathbf{K}_1]^{-1} & -[\mathbf{I} + \gamma \hat{\mathbf{D}}_1 \mathbf{K}_1]^{-1} \gamma \hat{\mathbf{D}}_1 \mathbf{k} \\ \mathbf{0}^\top & 1 \end{bmatrix} \begin{pmatrix} \mathbf{0} \\ \tilde{\alpha}_\ell \end{pmatrix} \\ &= \begin{pmatrix} -[\gamma^{-1}\hat{\mathbf{D}}_1^{-1} + \mathbf{K}_1]^{-1}\mathbf{k} \\ 1 \end{pmatrix} \tilde{\alpha}_\ell. \end{aligned}$$

The output of the kernel machine in the ℓ^{th} fold of the leave-one-out cross-validation procedure is then approximated by

$$\begin{aligned} \tilde{\mathbf{z}}^{(-\ell)} \approx \bar{\mathbf{z}}^{(-\ell)} &= \mathbf{K}\tilde{\boldsymbol{\alpha}} - \mathbf{K}\hat{\mathbf{H}}^{-1}\hat{\nabla} = \tilde{\mathbf{z}} - \begin{pmatrix} \mathbf{K}_1 & \mathbf{k} \\ \mathbf{k}^\top & K_{\ell\ell} \end{pmatrix} \begin{pmatrix} -[\gamma^{-1}\hat{\mathbf{D}}_1^{-1} + \mathbf{K}_1]^{-1}\mathbf{k} \\ 1 \end{pmatrix} \alpha_\ell \\ &= \tilde{\mathbf{z}} + \begin{pmatrix} \mathbf{K}_1[\gamma^{-1}\hat{\mathbf{D}}_1^{-1} + \mathbf{K}_1]^{-1}\mathbf{k} - \mathbf{k} \\ \mathbf{k}^\top[\gamma^{-1}\hat{\mathbf{D}}_1^{-1} + \mathbf{K}_1]^{-1}\mathbf{k} - K_{\ell\ell} \end{pmatrix} \alpha_\ell. \end{aligned}$$

The approximation to the ℓth output in the ℓth iteration of the procedure is then

$$\tilde{z}_\ell^{(-\ell)} \approx \bar{z}_\ell^{(-\ell)} = \tilde{z}_\ell - \left(K_{\ell\ell} - \mathbf{k}^\top \left[\frac{\mathbf{B}_1}{\gamma} + \mathbf{K}_1 \right]^{-1} \mathbf{k} \right) \alpha_\ell. \tag{21}$$

As a by-product of training the kernel machine on the full training sample, we have already evaluated

$$\mathbf{C}^{-1} = \begin{bmatrix} \mathbf{K}_1 + \frac{\mathbf{B}_1}{\gamma} & \mathbf{k} \\ \mathbf{k}^\top & K_{\ell\ell} + \frac{1}{\gamma\beta_\ell} \end{bmatrix}^{-1}.$$

Using the block matrix inversion lemma (19), the bottom right element of $\mathbf{C}^{-1}$ is given by

$$C_{\ell\ell}^{-1} = K_{\ell\ell} + \frac{1}{\gamma\beta_\ell} - \mathbf{k}^\top \left[\frac{\mathbf{B}_1}{\gamma} + \mathbf{K}_1 \right]^{-1} \mathbf{k},$$

substituting this into (21), we obtain

$$\bar{z}_\ell^{(-\ell)} = \tilde{z}_\ell - \left[\frac{1}{C_{\ell\ell}^{-1}} - \frac{1}{\gamma\beta_\ell} \right] \tilde{\alpha}_\ell.$$

Finally, noting that the training samples are interchangeable and substituting (10) and (11),

$$\bar{z}_i^{(-i)} = \eta_i - \frac{\tilde{\alpha}_i}{C_{ii}^{-1}},$$

which is identical to the result given in (20).

Appendix B: Analytic approximation of gradient information

Let $\Theta = \{\gamma, \theta_1, \ldots, \theta_d\}$ represent the set of hyper-parameters for a kernel logistic regression model. Near-optimal values for these hyper-parameters are chosen by minimizing an approximate leave-one-out cross-validation estimate of the true cross-entropy loss, i.e.

$$P(\Theta) = -\sum_{i=1}^{\ell} [t_i \log\{p_i^{(-i)}\} + (1 - t_i) \log\{1 - p_i^{(-i)}\}], \tag{22}$$

where

$$p_i^{(-i)} = \frac{1}{1 + \exp\{-\hat{z}_i^{(-i)}\}} \quad \text{and} \quad \hat{z}^{(-i)} = \eta_i - \frac{\alpha_i}{C_{ii}^{-1}}.$$

In order to implement an efficient gradient descent model selection procedure, we require the partial derivatives of $P(\Theta)$ with respect to the hyper-parameters. The coefficients of the quadratic approximation of the regularized loss, $\boldsymbol{\eta}$ and $\boldsymbol{\beta}$, Using the chain rule, we obtain,

$$\frac{P(\Theta)}{\partial \theta_i} = -\sum_{j=1}^{\ell} \frac{\partial P(\Theta)}{\partial \hat{z}_j^{(-j)}} \frac{\partial \hat{z}_j^{(-j)}}{\partial \theta_i}$$

where

$$\frac{\partial P(\Theta)}{\partial \hat{z}_j^{(-j)}} = p_j^{(-j)} - t_j \quad \text{and} \quad \frac{\partial \hat{z}_j^{(-j)}}{\partial \theta_i} = \frac{\alpha_j}{[C_{jj}^{-1}]^2} \frac{\partial C_{jj}^{-1}}{\partial \theta_j} - \frac{1}{C_{jj}^{-1}} \frac{\partial \alpha_j}{\partial \theta_i},$$

such that

$$\frac{P(\Theta)}{\partial \theta_i} = \sum_{j=1}^{\ell} \frac{\partial P(\Theta)}{\partial \hat{z}_j^{(-j)}} \frac{1}{C_{jj}^{-1}} \frac{\partial \alpha_j}{\partial \theta_i}, - \sum_{j=1}^{\ell} \frac{\partial P(\Theta)}{\partial \hat{z}_j^{(-j)}} \frac{\alpha_j}{[C_{jj}^{-1}]^2} \frac{\partial C_{jj}^{-1}}{\partial \theta_j}. \tag{23}$$

The parameters of a kernel logistic regression model, $[\boldsymbol{\alpha}^\top,\ b]^\top$, are given by a system of linear equations,

$$[\boldsymbol{\alpha}^\top\ b]^\top = \mathbf{C}^{-1}[\boldsymbol{\eta}^\top\ 0]^\top.$$

Using the following identity for the derivatives of the inverse of a matrix,

$$\frac{\partial \mathbf{C}^{-1}}{\partial \theta_i} = -\mathbf{C}^{-1} \frac{\partial \mathbf{C}}{\partial \theta_i} \mathbf{C}^{-1} \tag{24}$$

we obtain

$$\frac{\partial [\boldsymbol{\alpha}^\top\ b]^\top}{\partial \theta_i} = -\mathbf{C}^{-1} \frac{\partial \mathbf{C}}{\partial \theta_i} \mathbf{C}^{-1} [\boldsymbol{\eta}^\top\ 0]^\top = -\mathbf{C}^{-1} \frac{\mathbf{C}}{\partial \theta_i} [\boldsymbol{\alpha}^\top\ b]^\top. \tag{25}$$

The partial derivatives of $\mathbf{C}$ with respect to the regularization parameter, γ, and a kernel parameter, θ_i, are given by

$$\frac{\partial \mathbf{C}}{\partial \gamma} = \begin{bmatrix} \mathbf{B} & \mathbf{0} \\ \mathbf{0}^\top & 0 \end{bmatrix} \quad \text{and} \quad \frac{\partial \mathbf{C}}{\partial \theta_i} = \begin{bmatrix} \partial \mathbf{K}/\partial \theta_i & \mathbf{0} \\ \mathbf{0}^\top & 0 \end{bmatrix},$$

respectively. As (25) involves two matrix-vector products, the partial derivatives of the model parameters, and therefore the first summation in (23), can be computed with a complexity of $\mathcal{O}(\ell^2)$ operations per hyper-parameter. The second summation can be written as

$$-\sum_{j=1}^{\ell} \frac{\partial P(\Theta)}{\partial \hat{z}_j^{(-j)}} \frac{\alpha_j}{[C_{jj}^{-1}]^2} \frac{\partial C_{jj}^{-1}}{\partial \theta_j} = \text{Trace}\left\{\mathbf{C}^{-1}\frac{\partial \mathbf{C}}{\partial \theta_i}\mathbf{C}^{-1}\mathbf{D}\right\}$$

where

$$\mathbf{D} = \text{diag}\left(\frac{\partial P(\boldsymbol{\Theta})}{\partial \hat{z}_j^{(-j)}} \frac{\alpha_j}{[C_{jj}^{-1}]^2}\right).$$

Noting that Trace(**ABAD**) = Trace(**ADAB**) and defining $\mathbf{M} = \mathbf{C}^{-1}\mathbf{D}\mathbf{C}^{-1}$, where **D** is diagonal, we have

$$-\sum_{j=1}^{\ell} \frac{\partial P(\Theta)}{\partial \hat{z}_j^{(-j)}} \frac{\alpha_j}{[C_{jj}^{-1}]^2} \frac{\partial C_{jj}^{-1}}{\partial \theta_j} = \text{Trace}\left\{\mathbf{M}\frac{\partial \mathbf{C}}{\partial \theta_i}\right\}. \tag{26}$$

Therefore, provided we pre-compute **M**, then the partial derivatives of the model selection criterion with respect to all of the kernel hyper-parameters can be evaluated with a computational complexity of only $\mathcal{O}(\ell^3 + d\ell^2)$ operations. There exist a great variety of kernel functions, however for this study we adopt the isotropic Gaussian radial basis function kernel (1), for which the partial derivatives are given by

$$\frac{\partial \mathcal{K}(\mathbf{x}, \mathbf{x}')}{\partial \theta_1} = -\mathcal{K}(\mathbf{x}, \mathbf{x}')\|\mathbf{x} - \mathbf{x}'\|^2. \tag{27}$$

Since the regularization parameter, γ, and the scale parameters of a radial basis function kernel are strictly positive quantities, in order to permit the use of an unconstrained optimization procedure, we might adopt the parameterization $\tilde{\theta}_i = \log_2 \theta_i$, such that

$$\frac{\partial P(\Theta)}{\partial \tilde{\theta}_i} = \frac{\partial P(\Theta)}{\partial \theta_i}\frac{\partial \theta_i}{\partial \tilde{\theta}_i} \quad \text{where } \frac{\partial \theta_i}{\partial \tilde{\theta}_i} = \theta_i \log 2. \tag{28}$$

References

Allen, D. M. (1974). The relationship between variable selection and prediction. *Technometrics*, *16*, 125–127.

Anderson, E., Bai, Z., Bischof, C., Blackford, S., Demmel, J., Dongarra, J., Du Croz, J., Greenbaum, A., Hammarling, S., McKenney, A., & Sorensen, D. (1999). *LAPACK users' guide* (3rd ed.). Philadelphia: Society for Industrial and Applied Mathematics.

Bo, L., Wang, L., & Jiao, L. (2006). Feature scaling for kernel Fisher discriminant analysis using leave-one-out cross validation. *Neural Computation*, *18*(4), 961–978.

Boser, B. E., Guyon, I. M., & Vapnik, V. (1992). A training algorithm for optimal margin classifiers. In D. Haussler (Ed.), *Proceedings of the fifth annual ACM workshop on computational learning theory* (pp. 144–152), Pittsburgh, PA, July 1992.

Boyd, S., & Vandenberghe, L. (2004). *Convex optimization*. Cambridge: Cambridge University Press.

Brown, M. P. S., Grundy, W. N., Lin, D., Cristianini, N., Sugnet, C. W., Furey, T. S., Ares, M. Jr., & Haussler, D. (2000). Knowledge-based analysis of microarray gene expression data by using support vector machines. *Proceedings of the National Academy of Sciences*, *97*(1), 262–267.

Cawley, G. C., Janacek, G. J., & Talbot, N. L. C. (2007). Generalised kernel machines. In *Proceedings of the IEEE/INNS international joint conference on neural networks (IJCNN-07)*, Orlando, FL, USA, 12–17 August 2007.

Cawley, G. C., & Talbot, N. L. C. (2003). Efficient leave-one-out cross-validation of kernel Fisher discriminant classifiers. *Pattern Recognition*, *36*(11), 2585–2592.

Cawley, G. C., & Talbot, N. L. C. (2004a). Efficient model selection for kernel logistic regression. In *Proceedings of the 17th international conference on pattern recognition (ICPR-2004)* (Vol. 2, pp. 439–442), Cambridge, United Kingdom, 23–26 August 2004.

Cawley, G. C., & Talbot, N. L. C. (2004b). Fast leave-one-out cross-validation of sparse least-squares support vector machines. *Neural Networks*, *17*(10), 1467–1475.

Cawley, G. C., & Talbot, N. L. C. (2007). Preventing over-fitting in model selection via Bayesian regularization of the hyper-parameters. *Journal of Machine Learning Research*, *8*, 841–861.

Cawley, G. C., Talbot, N. L. C., Janacek, G. J., & Peck, M. W. (2006). Parametric accelerated life survival analysis using sparse Bayesian kernel learning methods. *IEEE Transactions on Neural Networks*, *17*(2), 471–481.

Chapelle, O. (2006). *Leave k out for kernel machines. unpublished research note*. 2 October 2006.

Chapelle, O. (2007). Training a support vector machine in the primal. *Neural Computation*, *19*(5), 1155–1178.

Chapelle, O., Vapnik, V., Bousquet, O., & Mukherjee, S. (2002). Choosing multiple parameters for support vector machines. *Machine Learning*, *46*(1), 131–159.

Cook, R. D., & Weisberg, S. (1982). *Monographs on statistics and applied probability. Residuals and influence in regression*. New York: Chapman and Hall.

Cortes, C., & Vapnik, V. (1995). Support vector networks. *Machine Learning*, *20*, 273–297.

Geman, S., Bienenstock, E., & Doursat, R. (1992). Neural networks and the bias/variance dilemma. *Neural Computation*, *4*(1), 1–58.

Golub, G. H., & Van Loan, C. F. (1996). *Matrix computations* (3rd. ed.). Baltimore: The Johns Hopkins University Press.

Green, P. J., & Silverman, B. W. (1994). *Monographs on statistics and applied probability: Vol. 58. Nonparametric regression and generalized linear models—a roughness penalty approach*. London: Chapman & Hall/CRC.

Kimeldorf, G. S., & Wahba, G. (1971). Some results on Tchebycheffian spline functions. *Journal of Mathematical Analysis and Applications*, *33*, 82–95.

Lachenbruch, P. A., & Mickey, M. R. (1968). Estimation of error rates in discriminant analysis. *Technometrics*, *10*(1), 1–11.

Luntz, A., & Brailovsky, V. (1969). On estimation of characters obtained in statistical procedure of recognition. *Techicheskaya Kibernetica*, *3* (in Russian).

Mercer, J. (1909). Functions of positive and negative type and their connection with the theory of integral equations. *Philosophical Transactions of the Royal Society of London, A*, *209*, 415–446.

Mika, S., Rätsch, G., Weston, J., Schölkopf, B., & Müller, K.-R. (1999). Fisher discriminant analysis with kernels. In *Neural networks for signal processing* (Vol. IX, pp. 41–48). New York: IEEE Press.

Mika, S., Rätsch, G., Weston, J., Schölkopf, B., Smola, A. J., & Müller, K.-R. (2000). Invariant feature extraction and classification in feature spaces. In S. A. Solla, T. K. Leen, & K.-R. Müller (Eds.), *Advances in neural information processing systems* (Vol. 12, pp. 526–532). Cambridge: MIT Press.

Mika, S., Rätsch, G., Weston, J., Schölkopf, B., Smola, A., & Müller, K.-R. (2003). Constructing descriptive and discriminative features: Rayleigh coefficients in kernel feature spaces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *25*(5), 623–628.

Minka, T. (2001). Expectation propagation for approximate Bayesian inference. In *Proceedings of uncertainty in artificial intelligence* (pp. 362–369).

Müller, K.-R., Mika, S., Rätsch, G., Tsuda, K., & Schölkopf, B. (2001). An introduction to kernel-based learning algorithms. *IEEE Transactions on Neural Networks*, *12*(2), 181–201.

Nabney, I. T. (1999). Efficient training of RBF networks for classification. In: *Proceedings of the ninth international conference on artificial neural networks* (Vol. 1, pp. 210–215), Edinburgh, United Kingdom, 7–10 September, 1999.

Nelder, J. A., & Mead, R. (1965). A simplex method for function minimization. *Computer Journal*, *7*, 308–313.

Opper, M., & Winther, O. (2000). Gaussian processes for classification: mean-field algorithms. *Neural Computation*, *12*(11), 2665–2684.

Rasmussen, C. E., & Williams, C. K. I. (2006). *Gaussian processes for machine learning. Adaptive computation and machine learning*. Cambridge: MIT Press.

Saunders, C., Gammermann, A., & Vovk, V. (1998). Ridge regression in dual variables. In J. Shavlik (Ed.), *Proceedings of the fifteenth international conference on machine learning (ICML-1998)*. San Mateo: Morgan Kaufmann.

Schölkopf, B., & Smola, A. J. (2002). *Learning with kernels—support vector machines, regularization, optimization and beyond*. Cambridge: MIT Press.

Schölkopf, B., Smola, A. J., & Müller, K. (1997). Kernel principal component analysis. In W. Gerstner, A. Germond, M. Hasler, & J.-D. Nicoud (Eds.), *Lecture notes in computer science: Vol. 1327. Proceedings of the international conference on artificial neural networks (ICANN-1997)* (pp. 583–588). Berlin: Springer.

Schölkopf, B., Herbrich, R., & Smola, A. J. (2002). A generalized representer theorem. In *Proceedings of the fourteenth international conference on computational learning theory* (pp. 416–426), Amsterdam, The Netherlands, 16–19 July 2002.

Seaks, T. (1972). SYMINV: an algorithm for the inversion of a positive definite matrix by the Cholesky decomposition. *Econometrica*, *40*(5), 961–962.

Shawe-Taylor, J., & Cristianini, N. (2004). *Kernel methods for pattern analysis*. Cambridge: Cambridge University Press.

Stone, M. (1974). Cross-validatory choice and assessment of statistical predictions. *Journal of the Royal Statistical Society, B*, *36*(1), 111–147.

Sundararajan, S., & Keerthi, S. S. (2001). Predictive approaches for choosing hyperparameters in Gaussian processes. *Neural Computation*, *13*(5), 1103–1118.

Suykens, J. A. K., De Brabanter, J., Lukas, L., & Vandewalle, J. (2002a). Weighted least squares support vector machines: robustness and sparse approximation. *Neurocomputing*, *48*(1–4), 85–105.

Suykens, J. A. K., Van Gestel, T., De Brabanter, J., De Moor, B., & Vanderwalle, J. (2002b). *Least squares support vector machines*. Singapore: World Scientific.

Tikhonov, A. N., & Arsenin, V. Y. (1977). *Solutions of ill-posed problems*. New York: Wiley.

Vapnik, V. (1998). *Statistical learning theory*. New York: Wiley.

Vapnik, V., & Chapelle, O. (2000). Bounds on error expectation for SVM. In: A. J. Smola, P. L. Bartlett, B. Schölkopf, & D. Schuurmans (Eds.), *Advances in large margin classifiers* (pp. 261–280).

Wahba, G. (1990). *Spline models for observational data*. Philadelphia: SIAM.

Weisberg, S. (1985). *Applied linear regression* (2nd ed.). New York: Wiley.

Williams, P. M. (1991). *A Marquardt algorithm for choosing the step-size in backpropagation learning with conjugate gradients*. Cognitive Science Research Paper CSRP-229, University of Sussex, Brighton, UK, February 1991.

Williams, C. K. I., & Barber, D. (1998). Bayesian classification with Gaussian processes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *20*(12), 1342–1351.

Zhu, J., & Hastie, T. (2005). Kernel logistic regression and the import vector machine. *Journal of Computational & Graphical Statistics*, *14*(1), 185–205.